Arabian Journal for Science and Engineering (2018) 43:7253–7264
https://doi.org/10.1007/s13369-017-3001-5

RESEARCH ARTICLE - COMPUTER ENGINEERING AND COMPUTER SCIENCE

Greedy algorithms for sparse signal recovery based on temporally correlated experimental data in WSNs

Poonam Goyal[1] · **Brahmjit Singh**[1]

Received: 3 April 2017 / Accepted: 23 November 2017 / Published online: 12 January 2018

Abstract

In wireless sensor networks (WSNs), thousands of sensor nodes are deployed to measure various environmental parameters such as temperature, light intensity, humidity and air pressure. All living beings can sense the variation in these parameters; therefore, these parameters are termed as natural signals. The natural signals are highly correlated in time and space; therefore, it can be compressed significantly to achieve the low sampling rates. The correlation property of natural signals is exploited here to compress/decompress these signals for reducing the transmission cost of the networks. The real-time temperature signal is measured using national instruments (NI) WSN platform, which is used for analysis purpose. The signal at first is transformed into sparse signal and then compressed. The compressed signal is transmitted to the receiver, where it is decoded into original sparse signal using algorithms based on greedy iterative approaches, i.e. orthogonal matching pursuit (OMP), stagewise orthogonal matching pursuit (StOMP) and generalized OMP (gOMP). The most popular greedy algorithm, OMP, is compared with StOMP and gOMP. The performance is analysed quantitatively in terms of peak signal-to-noise ratio, root-mean-squared error and execution speed of these greedy algorithms. It is demonstrated through simulation, the computational speed of StOMP and gOMP is much better than OMP, and also, the sparse signal is recovered with accuracy approximately equal to OMP.

Keywords Compressed sensing · Wireless sensor networks · Orthogonal matching pursuit · Sparse signal · Peak signal-to-noise ratio · Execution time

1 Introduction

Sparse signal recovery refers to the problem of recovering the sparse signals using few linearly transformed measurements that possess incoherence properties. Sparse signals have a few nonzero coefficients in a suitable transform basis or dictionary. Sparse signal recovery is also known as compressed sensing or compressive sampling [1]. Compressed sensing is being widely used in the areas like image processing, wireless communication, signal processing, cognitive radios, geophysics, astronomy. [2]. These applications involve high-dimensional signals, which require the large memory space and bandwidth in order to store and transmit them, respectively. This leads to inefficient use of available resources. Compressed sensing removes the constraints of limited bandwidth and storage space by encoding the large-dimensional signals into smaller dimensions and then recovering the original signals from their compressed form [3]. This theory was given by Donoho [4], Candès and Wakin [5]. In compressed sensing, a sparse signal $x \in R^N$ is sampled and compressed into a set of M measurements, whereas M is much smaller than N. If these measurements are taken in an appropriate order, then original sparse signal x can be possibly recovered using this small set of measurements [6].

Compressed sensing has attracted much attention in recent years for wireless sensor networks (WSNs) where measurement capabilities are limited due to energy constraints. WSNs are distributed networks of thousands of sensor nodes with the capabilities of sensing, processing and communication with the sink node and other sensor nodes also. These networks are being widely used in the areas like environmental monitoring, building automation, health, industrial monitoring, smart homes. [7]. Due to the implementation of sensor

✉ Poonam Goyal
poonamg1989@gmail.com

Brahmjit Singh
brahmjit@nitkkr.ac.in

1 Department of Electronics and Communication Engineering, National Institute of Technology Kurukshetra, Haryana, India

nodes on the large scale, the energy expenditure and transmission cost of the network increase to a great extent. Different kinds of routing and data fusion protocols have been proposed in the literature to reduce the energy consumption and transmission overhead [8–11]. In this article, use of compressed sensing is presented in WSNs to reduce the volume of data transmission within the network which further decreases the energy consumption of the network also.

Compressed sensing transforms a discrete time signal x of size N, i.e. $x \in R^N$ into a K sparse signal using a discrete cosine transform (DCT) domain matrix, Ψ, $(\Psi \in R^{N \times N})$ according to Eq. (1).

$$x = \Psi\alpha \tag{1}$$

where α is a column vector of transform coefficients. If $\alpha \in R^N$ has K elements of significant values such that rest of $(N - K)$ elements can be rejected without any substantial loss and $K \ll N$, then α is known as K sparse representation of original signal x in DCT domain. The matrix Ψ can be referred as the basis matrix. Compressed sensing encodes K sparse signal by computing a measurement vector y of dimension M, where $K < M < N$. As x has sparse representation with respect to basis Ψ, y is expressed as

$$y = \Phi x = \mathrm{A}\alpha \tag{2}$$

where Φ is known as measurement matrix as $\Phi \in R^{M \times N}$ and $\mathrm{A} \in R^{M \times N} = \Phi\Psi$. The measurement vector y is decoded into original signal x by determining the transform coefficient vector using $y = A\alpha$ where A is a rectangular matrix $\mathrm{A} \in R^{M \times N} = \Phi\Psi$ and Φ, Ψ matrices are known in advance [12,13]. The reconstruction of original signal x from a few random projections is an ill-posed problem; therefore, signal sparsity must be known in prior to recover x using $M \ll N$ projections only [14]. In compressed sensing, various sparse signal recovery algorithms have been presented to reconstruct the sparse signal from a few measurements. One of such algorithms is l_0 minimization as expressed in Eq. (3).

$$\min_{\tilde{\alpha}} \|\tilde{\alpha}\|_0 \quad \text{s.t.} \quad A\tilde{\alpha} = y \tag{3}$$

Unfortunately, this algorithm is very difficult in practice and is NP hard in general. Therefore, alternatives to NP hard problem are explored and then l_0 optimization algorithm replaced by l_1 minimization approach [15]. l_1 minimization algorithm is based on linear programming techniques and imposes the condition of restricted isometry property (RIP) on the measurement matrix A to obtain a unique sparsest solution. For the condition of RIP, there must exist a restricted isometry constant (RIC) δ_K, $0 < \delta_K < 1$ such that

$$(1 - \delta_K)\, \|\alpha\|_2^2 \le \|A\alpha\|_2^2 \le (1 + \delta_K)\, \|\alpha\|_2^2 \tag{4}$$

for all entries of transform coefficient vector, α [16]. However, l_1 minimization approach provides the strong guarantees of recovered signal yet infeasible for many applications due to their computational complexity and polynomial run time. Therefore, the decoding algorithms with strong signal recovery having less computational time become of critical importance. To solve this problem, a family of greedy iterative algorithms have been proposed, e.g. matching pursuit (MP) [17], orthogonal matching pursuit (OMP) [18], stagewise OMP (StOMP) [19], subspace pursuit (SP) [20], compressive sampling matching pursuit (CoSaMP) [21]. Among all these greedy algorithms, OMP is most widely used sparse signal recovery algorithm due to its simplicity and competitive performance. OMP algorithm iterates K times where K is the sparsity of signal to be recovered. In each iteration, a single column is selected from the measurement matrix on the basis of its correlation with the residual signal. In this way, computational time of OMP algorithm directly depends upon the sparsity of signal. Various efforts have been done to reduce the number of iterations of OMP algorithm with the aim of increasing the execution speed of the algorithm. Methods are proposed which can identify more than one indices in each iteration to increase the computational efficiency of OMP algorithm. These methods are all variants of OMP such as StOMP, SP and CoSaMP. [22]. These algorithms work on the modification of identification step of OMP in different ways. For example, in StOMP, multiple indices are chosen in each iteration whose correlation magnitudes are greater than a specified threshold value. In SP and CoSaMP, multiple indices are selected in the identification step, and after that, a large portion of selected indices is pruned to polish the identification step. Recently, an algorithm has been proposed in [23] with the objective of enhancing the recovery performance in less computational time in comparison with OMP. It is termed as generalized OMP (gOMP), also known as orthogonal super greedy algorithm (OSGA) [24] or orthogonal multi-matching pursuit (OMMA) [25]. Here, in this paper, a comparison is made between OMP, StOMP and gOMP to find out whichever algorithm provides fast and satisfactory sparse signal recovery. The comparative analysis is performed in terms of peak signal-to-noise ratio (PSNR), root-mean- squared error (RMSE) and execution time of the algorithms.

For comparison, the sparse signal is generated by representing the temporally correlated temperature signal into discrete cosine transform (DCT) domain. The main objectives of this work are as follows:

1. To measure real-time temperature data using national instruments (NI) WSN platform.
2. To apply compressed sensing on these measured values by compressing them using measurement matrix at the encoder. At the receiver end, to retrieve the origi-

nal sparse signal from its compressed form, sparse signal recovery algorithms, i.e. OMP, StOMP and gOMP, are implemented sequentially.

3. To evaluate the performance of recovery algorithms on the basis of the efficacy of recovered signal and execution speed.

The rest of paper is structured as follows. Section 2 describes the experimental set-up needed to record the temperature signal using NI WSN hardware followed by the compressive encoder and decoder in Sects. 2.2 and 2.3, respectively. OMP, StOMP and gOMP all are explained stepwise including their advantages and disadvantages. The performance parameters required to differentiate the recovery algorithms are depicted in Sect. 3 accompanied by the conclusion in Sect. 4.

2 Sparse signal recovery in WSNs: a case study

In environmental monitoring WSNs, the sensor nodes are deployed to measure the environmental parameters such as light intensity, temperature, air pressure, wind direction and humidity. The measurements of these parameters are highly correlated in time and space because many sensors monitor the same spatial region over varying sample intervals. Due to the deployment of a large number of sensor nodes for monitoring, same geographic area leads to spatial correlation among sensor measurements and slow variation with time caused these signals to be temporally correlated also [26]. The correlation of one-dimensional signal $x_i^{(k)}$ sensed by a node with the same signal shifted by m time samples, i.e. $x_i^{(k+m)}$, averaged for all the N signals of $X^{(k)} \in \mathbb{R}^N$ is known as temporal correlation $\rho_m\left(X^{(.)}\right)$ [27], which is defined as

$$\rho_m\left(X^{(.)}\right) = \sum_{i=1}^{N} \frac{1}{N} \frac{\sum_{k=1}^{K_T}\left(x_i^{(k)} - E\left[x_i\right]\right)\left(x_i^{(k+m)} - E\left[x_i\right]\right)}{K_T \sigma_{x_i}^2} \tag{5}$$

where time varies from $k = 1, \ldots, K_T$.

Compressed sensing exploits the spatial–temporal correlations within the sensor measurements to make them highly compressible, hence achieving much lower sampling rates. In further subsections, it is explained in detail that how the temperature values are read using NI WSN hardware kit and LabVIEW and further compressed using CS encoder and recovered via iterative greedy algorithms.

2.1 Experimental set-up

The testbed for real-time temperature measurement consists of NI WSN-9791 Ethernet gateway, NI WSN-3212 programmable thermocouple input node, one set of 4 AA batteries for NI WSN measurement node, J-type thermocouple, Ethernet cable, graphical NI LabVIEW software installed on the desktop and power supply for WSN-9791. The various components and their connections are displayed in Figs. 1 and 2, respectively.

As shown in Fig. 2, a network is established between WSN gateway and WSN measurement node connected with a J-type thermocouple, using IEEE 802.15.4 communication

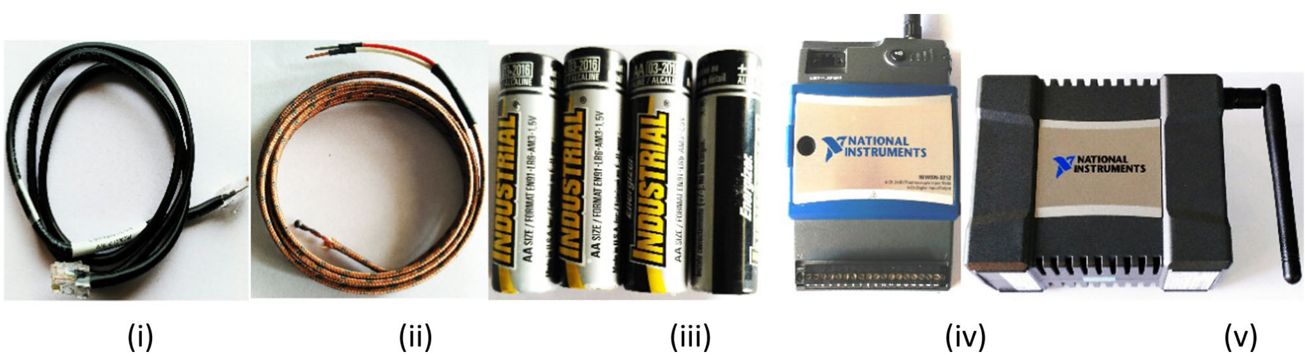


Fig. 1 **i** Ethernet cable; **ii** thermocouple sensor; **iii** AA battery; **iv** NI WSN 3212 thermocouple measurement node; **v** NI WSN 9791 Ethernet gateway

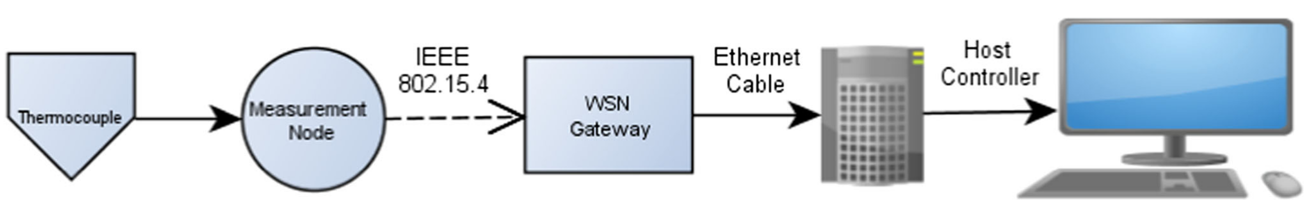


Fig. 2 WSN set-up for temperature measurement

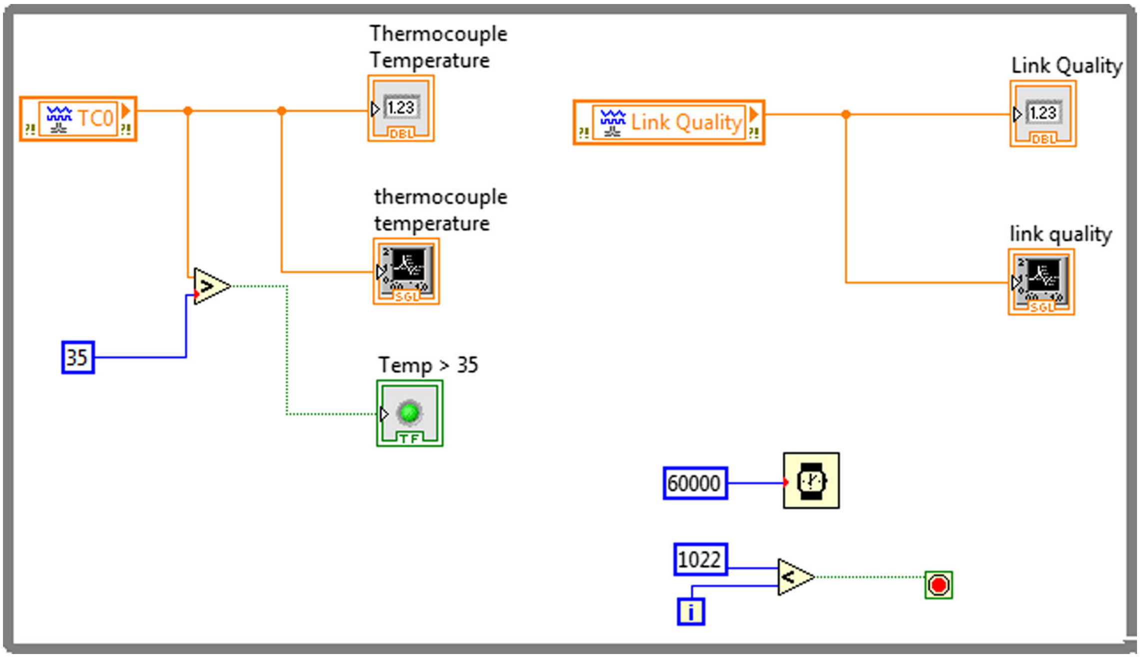


Fig. 3 Thermocouple temperature and link quality measurements using LabVIEW

protocol. The WSN gateway is attached using Ethernet cable to the desktop loaded with LabVIEW. The WSN measurement node and gateway are configured in LabVIEW using NI MAX (National Instruments Measurement and Automation Explorer) software. After detecting the WSN node in LabVIEW, each and every pin of measurement node can be programmed. The thermocouple temperature and link quality pins are configured in the following way as shown in Fig. 3.

2.2 Compressive sensing encoder

In present WSNs, the temperature samples are measured at an interval of 60 s. These samples are found highly temporally correlated with the temporal correlation of 0.9, calculated according to Eq. (5). These samples are sparsified using a transform domain matrix Ψ. The matrix Ψ must satisfy the following two criteria: (1) it must be able to sufficiently sparse the signal, and (2) it must be incoherent with corresponding measurement or sensing matrix Φ [28]. Here, discrete cosine transform (DCT) matrix is used for the same.

DCT matrix is also known as representation matrix or basis matrix. DCT matrix uses dynamic thresholding approach for determining the K sparse elements. The mathematical expression for transform matrix of DCT is given below.

$$\mathrm{DCT}\,(N \times N) = \left[\psi_{i,j} = \mathrm{C}\cos\left(i\,(1+2j)\,\frac{\pi}{2N}\right)\right] \tag{6}$$

where $i \in \{0, \ldots, N-1\}$ and $j \in \{0, \ldots, N-1\}$ represent the row and column numbers of the matrix. C is a constant, defined as follows

$$\mathrm{C} = \begin{cases} \sqrt{\frac{1}{N}}, & \text{for} \quad i = 0 \\ \sqrt{\frac{2}{N}}, & \text{for} \quad i \neq 0 \end{cases} \tag{7}$$

DCT matrix is orthogonal; hence, its inverse is obtained by taking the transpose of the matrix, IDCT=DCT^T. DCT matrix makes a signal sparse by concentrating most of the information in the low-frequency components. Therefore, high-frequency components have very less information; hence, they are eliminated without any significant loss [13].

Another matrix, known as measurement matrix, $\Phi \in R^{M \times N}$, is used for compressing the signal during the signal acquisition step, i.e. $y = \Phi x$. The matrix must be structured in such a way that it consists of least number of nonzeros in order to reduce the complexity of encoding. Different kinds of measurement matrices have been proposed in the literature, for example Gaussian random matrix, Fourier matrix, scrambled Fourier (SF) matrix, partial noiselet (PN) matrix. [29]. All these matrices suffer from the limitations of large computational complexity (because of large number of nonzero entries) and storage space. The simplest measurement matrix proposed by Ravelomanantsoa [30] is used for compressing the data in the presented work. The matrix is called as deterministic binary block diagonal (DBBD) and is expressed as

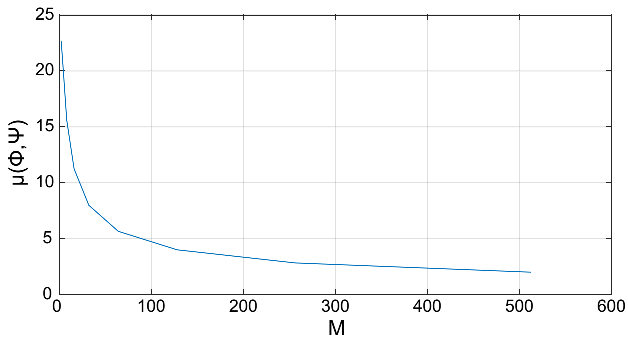


Fig. 4 Coherence between Φ and Ψ matrices

$$\Phi_{DBBD} = \begin{bmatrix} \overbrace{[1\cdots 1]}^{m} & 0 & 0 & 0 \\ 0 & \overbrace{[1\cdots 1]}^{m} & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & \overbrace{[1\cdots 1]}^{m} \end{bmatrix} \quad (8)$$

The blocks of this diagonal matrix are identical and contain $m = N/M$, elements each, where M and N are the number of rows and columns of measurement matrix, respectively. According to second criteria for basis matrix Ψ, it must be incoherent with the measurement matrix Φ. It means that each and every element of measurement matrix Φ must not be correlated with the elements of basis matrix Ψ. The coherence μ among two matrices is expressed below

$$\mu(\Phi, \Psi) = \sqrt{N} \max_{i,j} \frac{|\langle \phi_i, \psi_j \rangle|}{\|\phi_i\|_2 \|\psi_j\|_2} \quad (9)$$

where $\phi_{i\in\{1,\ldots,M\}}$ and $\psi_{j\in\{1,\ldots,N\}}$, respectively, represent the row vectors of Φ and column vectors of Ψ. Since $\mu \in [1, \sqrt{N}]$, for matrices Φ and Ψ to be incoherent, μ must be closer to one [12,16]. As the value of μ approaches to $\sqrt{N}$, Φ and Ψ matrices become less incoherent with each other and lead to poor signal recovery. The coherence between DBBD and DCT matrices is shown in Fig. 4. The coherence between two matrices decreases as the number of rows M in the measurement matrix increases which makes these two matrices more uncorrelated w.r.t. each other.

2.3 Compressive sensing decoder

In this subsection, the algorithms based on a greedy iterative approach for sparse signal recovery are described. These algorithms are required at the sink node to decode the compressed signal. The sink node is free from the constraints of limited bandwidth and energy. Greedy algorithms compute the support of the sparse signal iteratively. After calculating the signal support, the pseudoinverse of the matrix confined to the selected columns is used to reconstruct the sparse signal. These algorithms feature the fast signal recovery but face some disputes also.

2.3.1 Orthogonal matching pursuit

To recover the sparse signal α, we need to determine which columns of matrix A have significant contribution in the measurement vector y. To solve this problem, the columns from matrix A are selected in a greedy order using OMP algorithm [18]. OMP is an iterative greedy algorithm in which a single column is selected from A which found to be most correlated with the residue part of the signal y at each step. The selected column is then added to the set of already selected columns, hence framing a matrix A_{opt}. The pseudoinverse of matrix A_{opt} is calculated to solve the least square fit, hence obtaining a new sparse signal estimate. The residual is updated iteratively by subtracting the contribution of selected columns from the measurement vector, hence finding a global optimal solution for the signal α. This algorithm was proposed by Mallat and Zhang [31]. The steps of OMP are described in Table 1.

Since OMP algorithm iterates K times to compute the residual correlations and solve a least square problem for the sparse signal approximation, it requires large run time. In addition to this, algorithm builds up the optimal set one element at a time. The efficient implementation of OMP would necessarily require Cholesky factorization of optimal set, thereby reducing the complexity of solving the least square problem. In K steps, OMP would take at most $4K^3/3 + KMN + O(N)$ arithmetic operations. In worst case, where no sparsity is assumed, OMP will take M steps for performing the $4M^3/3 + M^2N + O(N)$ operations [19].

2.3.2 Stagewise orthogonal matching pursuit

Unlike OMP, instead of selecting the largest component of measurement vector y, in StOMP multiple coordinates are selected whose correlation magnitudes are above a specified threshold value. After that, least square problem is solved similar to OMP and residual is updated iteratively. In addition to this, StOMP completes in a few iterations (e.g. 5), while OMP can take many (e.g. K) where K is the sparsity level of signal to be recovered. Therefore, StOMP must run much faster than OMP. StOMP was developed and analysed by Donoho et al. [19]. StOMP can be executed in the following steps as shown in Table 2.

The formal noise level $\sigma_S = \|r_S\|_2 / \sqrt{M}$, and threshold parameter t_S takes values in the range $2 \le t_S \le 3$. The threshold value has been designed in such a way that many coefficients can enter at each stage and algorithm quits after a

Table 1 OMP algorithm

INPUT

- An $M \times N$ dimensional matrix A.
- An M dimensional measurement vector y.
- The sparsity level K for the sparse signal α.

OUTPUT

- Signal estimate $\tilde{\alpha}$ in $\mathbb{R}^N$ of the original signal.
- A set Λ_K containing K elements from $\{1, \ldots, N\}$.
- An M dimensional approximation b of the measurement vector y.
- An M dimensional residual $r = y - b$.

STEPS

1) Initialize the residual $r_0 = y$, the index set $\Lambda_0 = \emptyset$ and the iteration counter $t = 1$.
2) Find the new coordinate λ_t that solves the optimization problem

$$\lambda_t = \arg\max_{j=1,\ldots,N} \left| \langle r_{t-1}, A_j \rangle \right|$$

If the maximum occurs for multiple indices, break the tie deterministically.

3) Intensify the index set $\Lambda_t = \Lambda_{t-1} \cup \{\lambda_t\}$ and the matrix of chosen columns $A_{\mathrm{opt(t)}} = \left[A_{\mathrm{opt(t-1)}} \; A_{\lambda_t}\right]$ and A_{opt} is an empty matrix initially. After t iterations, A_{opt} will be the updated matrix of dimensions $M \times K$, i.e. append the selected column vector of A to the augmented matrix A_{opt}.
4) Solve a least square problem to obtain a new sparse signal estimate

$$\alpha_t = \arg\min_{\alpha} \left\| \left(A_{\mathrm{opt(t)}}\alpha\right) - y \right\|_2$$

The pseudoinverse of matrix A_{opt} is calculated using the following equation to calculate the new signal approximant.

$$\alpha_t = \left(A_{\mathrm{opt}}^{\mathrm{T}} A_{\mathrm{opt}}\right)^{-1} A_{\mathrm{opt}}^{\mathrm{T}} y$$

5) Update the residual

$$b_t = A_{\mathrm{opt(t)}}\alpha_t$$
$$r_t = y - b_t$$

Increment t, and return to step 2 if $t < K$.

6) After calculating the sparse coefficient vector $\tilde{\alpha}$, the real signal is estimated by multiplying the $\tilde{\alpha}$ with the Ψ_{IDCT} matrix.

$$\tilde{x} = \Psi_{\mathrm{IDCT}}\tilde{\alpha}$$

NOTE: at the end of t iterations, α_t is denoted by $\tilde{\alpha}$ which has the non zero elements of α_t at the locations (Λ_t) corresponding to the respective column number from the matrix A and rest of the elements will be zero making it a sparse coefficient vector $\tilde{\alpha}$ having the dimensions of $N \times 1$.

In Table 1, firstly, the residual, index set and iteration counter are initialized. Then an index from matrix A is selected, which found maximally correlated with the residual signal. The selected index is then added to the set Λ_t, and its size increases as the number of iterations increases. The selected indices from matrix A frame a submatrix A_{opt} which is used for solving the least square problem in the next step. The least square problem is solved to compute the sparse signal using pseudoinverse of matrix A_{opt}. After calculating the sparse signal, residual is updated and the process is repeated iteratively. At last, sparse signal is multiplied with inverse DCT matrix to obtain the original signal

fixed number of iterations. The number of coefficients entering each stage should be equal to K, but sometimes, it falls below K due to missed detections or false discoveries which degrades the performance of algorithm. The main disadvantage of this algorithm is determining an appropriate value of threshold as different threshold values could lead to different results.

StOMP algorithm solves the least square problem via conjugate gradient (CG) solver requiring at most $S\,(v + 2)\,MN + \mathrm{O}\,(N)$ arithmetic operations in S stages, where v is the number of CG iterations [19]. As StOMP has modest storage requirements at any step of algorithm execution, there is only need to store the current estimate α_S, current residual vector r_S and the current active set Λ_S. This makes StOMP feasible for very large scale sparse applications.

2.3.3 Generalized orthogonal matching pursuit

gOMP is simply a generalization of OMP algorithm. This algorithm is designed with the aim of reducing the com-

Table 2 StOMP algorithm

INPUT

- An $M \times N$ dimensional matrix A.
- An M dimensional measurement vector y.
- The sparsity level K for the sparse signal α.

OUTPUT

- Signal estimate $\tilde{\alpha}$ in $\mathbb{R}^N$ of the original signal.
- A set Λ_K containing less than or equal to K elements from $\{1, \ldots, N\}$.
- An M dimensional approximation b of the measurement vector y.
- An M dimensional residual $r = y - b$.

STEPS

1) Initialize the residual $r_0 = y$, the index set $\Lambda_0 = \emptyset$ and the iteration counter $S = 1$.
2) At the Sth stage, matched filtering is applied to the current residual to find out the residual correlations.
$$c_S = A^T r_{S-1}$$
3) Find the set of coordinates λ_S, whose values are above the specified threshold value.
$$\lambda_S = \{\, j : |c_S(j)| > t_S \sigma_S \,\}$$
here, σ_S is a formal noise level and t_S is a threshold parameter.
4) Augment the index set $\Lambda_S = \Lambda_{S-1} \cup \{\lambda_S\}$ and the matrix of chosen columns $A_{\mathrm{opt(S)}} = \left[A_{\mathrm{opt(S-1)}} \;\; A_{\lambda_S}\right]$ and A_{opt} is an empty matrix initially. After S iterations, A_{opt} will be the updated matrix of dimensions less than or equal to $M \times K$, i.e. append the selected column vectors of A to the augmented matrix A_{opt}.
5) Solve a least square problem to obtain a new sparse signal estimate
$$\alpha_S = \arg\min_{\alpha} \left\| \left(A_{\mathrm{opt(S)}} \alpha\right) - y \right\|_2$$
The pseudoinverse of matrix A_{opt} is estimated using the following equation to calculate the new signal approximant.
$$\alpha_S = \left(A_{\mathrm{opt}}^{\mathrm{T}} A_{\mathrm{opt}}\right)^{-1} A_{\mathrm{opt}}^{\mathrm{T}} y$$
6) Update the residual
$$b_S = A_{\mathrm{opt(S)}} \alpha_S$$
$$r_S = y - b_S$$
Check the stopping condition, if it is not time to stop then set $S = S + 1$ and repeat all the steps.
7) After calculating the sparse coefficient vector $\tilde{\alpha}$, the real signal is estimated by multiplying the $\tilde{\alpha}$ with the Ψ_{IDCT} matrix.
$$\tilde{x} = \Psi_{\mathrm{IDCT}} \tilde{\alpha}$$

NOTE: at the end of S iterations, α_S is denoted by $\tilde{\alpha}$ which has the non zero elements of α_S at the locations (Λ_S) corresponding to the respective column number from the matrix A and rest of the elements will be zero making it a sparse coefficient vector $\tilde{\alpha}$ having the dimensions of $N \times 1$

In StOMP algorithm, residual, index set and iteration counter are initialized in the first step. The correlation between the columns of matrix A and residual is calculated. In third step, multiple indices from matrix A are selected having correlation values above the threshold value, and then, selected indices are added to the index set Λ_S. The indices in the set Λ_S construct the matrix A_{opt} which is further used for computing the sparse signal by solving the least square problem. The residual is updated in the next step, and all these steps are repeated iteratively to approximate the original signal

putational cost of OMP. It only differs with OMP in the identification step by selecting n columns in each iteration for estimating the support of sparse signal. The value of n for gOMP is bounded between $n \leq K \; and \; n \leq M/K$, whereas OMP is treated as special case of gOMP with the value of $n = 1$ [23,32,33] Since OMP algorithm requires that all the chosen indices are "correct" (which is in actual support of sparse signal) for acceptable sparse signal recovery. However, this is not possible. In gOMP, if one "correct" index is chosen among the multiple selected indices, then maximum iterations will be equal to K as in OMP. However, there is probability for the selection of multiple "correct" indices at a

Table 3 gOMP algorithm

INPUT

- An $M \times N$ dimensional matrix A.
- An M dimensional measurement vector y.
- The sparsity level K for the sparse signal α.
- Number of indices for each step, n ($n \le K$ and $n \le M/K$).

OUTPUT

- Signal estimate $\tilde{\alpha}$ in $\mathbb{R}^N$ of the original signal.
- A set Λ_K containing K elements from $\{1, \ldots, N\}$.
- Estimated signal $\alpha = \arg\min_{\alpha: supp(\alpha)=\Lambda_k} \|A\alpha - y\|_2$

STEPS

1) Initialize the residual $r_0 = y$, the index set $\Lambda_0 = \emptyset$ and the iteration counter $k = 0$.
2) Increment the counter $k = k + 1$.
3) Select the coordinates $\{A_{(i)}\}_{i=1,2,\ldots,n}$ corresponding to n largest entries in correlation magnitude $A^T r_{k-1}$.
4) Intensify the index set $\Lambda_k = \Lambda_{k-1} \cup \{A(1), A(2), \ldots, A(n)\}$.
5) Solve a least square problem to obtain sparse signal estimate
$$\alpha_{\Lambda_k} = \arg\min_{\alpha} \left\| A_{\Lambda_k}\alpha - y \right\|_2$$
6) Update the residual
$$b_k = A_{\Lambda_k}\alpha_{\Lambda_k}$$
$$r_k = y - b_k$$
Repeat the steps 2 to 6 until $k < min\{K, M/n\}$ and $\|r_k\|_2 > \epsilon$.
7) After calculating the sparse coefficient vector $\tilde{\alpha}$, the real signal is estimated by multiplying the $\tilde{\alpha}$ with the Ψ_{IDCT} matrix.
$$\tilde{x} = \Psi_{\text{IDCT}}\tilde{\alpha}$$

NOTE: at the end of k iterations, α_{Λ_k} is denoted by $\tilde{\alpha}$ which has the non zero elements of α_{Λ_k} at the locations (Λ_k) corresponding to the respective column number from the matrix A and rest of the elements will be zero making it a sparse coefficient vector $\tilde{\alpha}$ having the dimensions of $N \times 1$.

In Table 3 for gOMP algorithm, with the initialization of residual, index set and iteration counter, the correlation between the columns of matrix A and residual is calculated. The correlation magnitudes are sorted in the descending order, and indices corresponding to n largest entries are selected from matrix A and added to the set Λ_k. Then, least square problem is solved to approximate the sparse signal. The process is repeated until the conditions of termination are satisfied

time; then, number of iterations will certainly be less than K. The iterations required in gOMP algorithm are denoted by k, and its value should be $k < min\left\{K, M/n\right\}$. Like OMP and StOMP, the complexity of gOMP is directly dependent upon the number of iterations, i.e. $2kMN + \left(2n^2 + n\right)k^2M$. Less the number of iterations, lesser will be the computational cost of the algorithm.

3 Results and analysis

The simulation is performed on a system supporting an Intel CORE i3 processor and 4 GB memory space with MATLAB 8.5. A temperature signal measured using NI WSN-3212 thermocouple node by keeping in an open environment is analysed. The signal is represented as a sparse signal in DCT domain and further compressed using Φ_{DBBD} measurement matrix during signal acquisition step. The compressed signal is converted into original sparse signal using OMP, StOMP and gOMP sequentially. The comparative analysis is performed on the basis of RMSE, PSNR and execution time.

3.1 Performance parameters

The performance of sparse signal recovery algorithms is evaluated using the following parameters.

Root-mean-squared error (RMSE)—RMSE values are calculated using the following mathematical expression

$$\text{RMSE} = \sqrt{\frac{\sum_1^N (x - \tilde{x})^2}{N}} \tag{10}$$

where x and $\tilde{x}$ are the original and reconstructed signal, respectively.

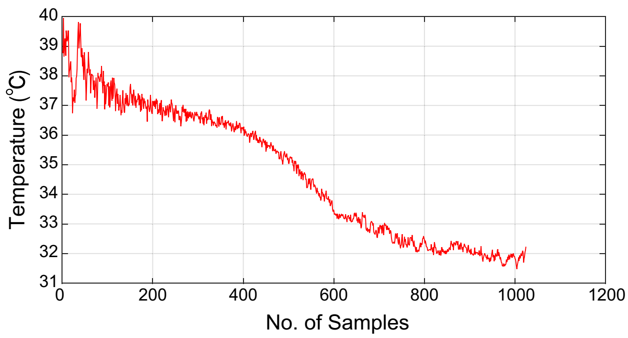


Fig. 5 Original temperature signal

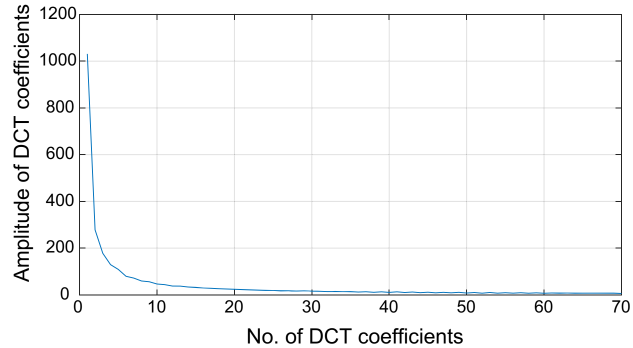


Fig. 6 Sparsity analysis in temporally correlated data

PSNR—The distortion between original signal and recovered signal is measured using peak signal-to-noise ratio as [30]

$$\text{PSNR [dB]} = 10\log_{10}\left(\frac{\max x^2}{\frac{1}{N}\sum_0^{N-1}(x-\tilde{x})^2}\right) \quad (11)$$

Compression ratio (CR)—For various reconstruction algorithms, CR will be different. For example, in OMP and gOMP, the number of samples required to reconstruct the signal of length N is equal to K; hence, CR is

$$\text{CR} = \frac{N}{K} \quad (12)$$

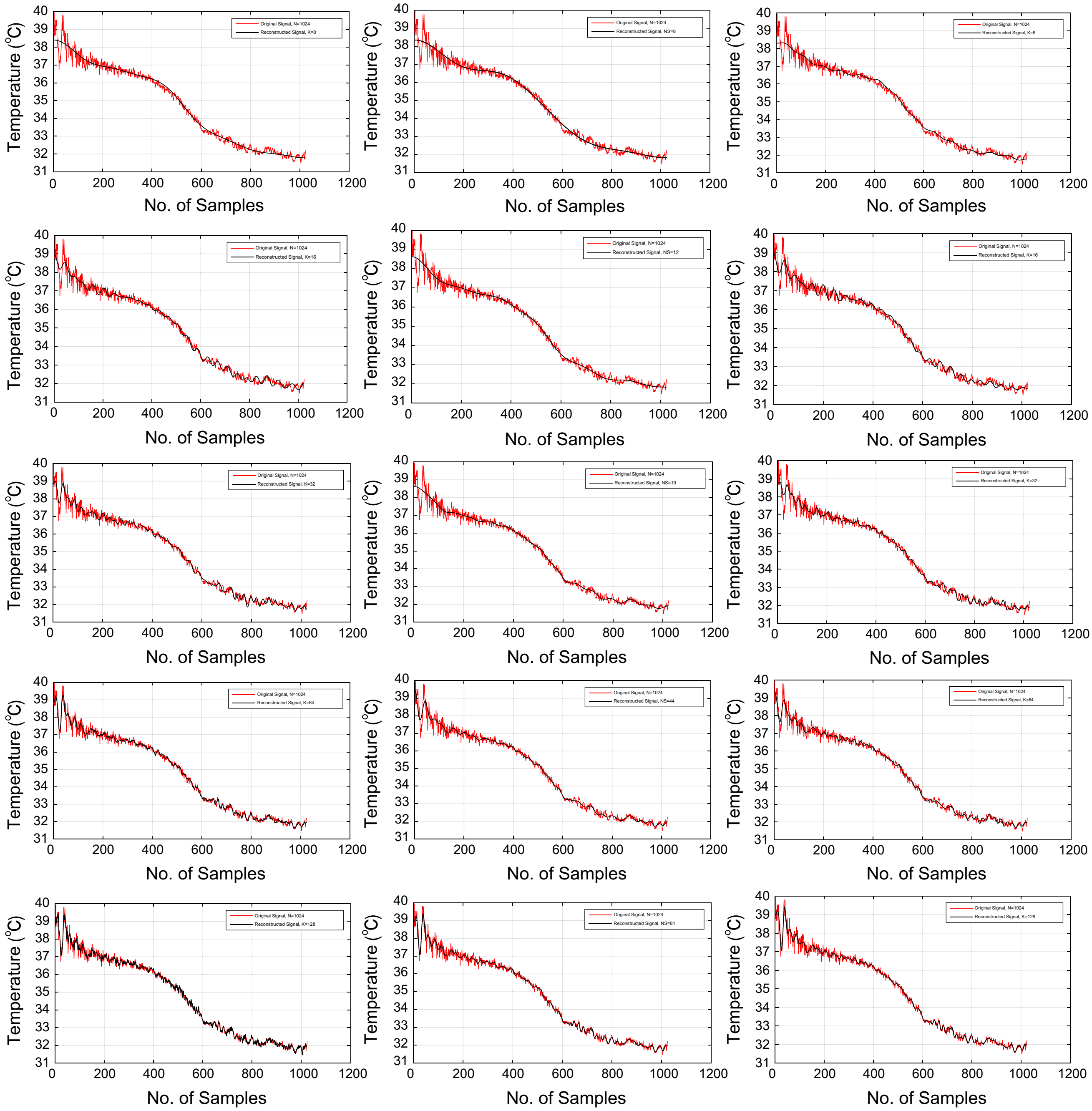
Although, in StOMP algorithm, the samples needed to recover the original signal may be equal to or less than K (due to false alarms) depending upon the applications, therefore, there is not any standard mathematical formula to calculate CR in case of StOMP.

3.2 Performance analysis

The WSN measurement node equipped with a thermocouple at its input pins was placed in an open environment to measure the temperature. The temperature values available at the output pins of measurement node are visualized in LabVIEW and then exported to MATLAB for further analysis. The original temperature signal is displayed in Fig. 5 where 1024 temperature samples are shown.

Sparsity Analysis Since the signal is highly temporally correlated, it becomes enough sparse when represented into a transform domain. Here, DCT is applied for making the signal sparse. DCT operates by concentrating all information in low-frequency components and decreases the contribution of higher-frequency components; hence, they are eliminated. In this way, the signal is made as sparse by retaining a few coefficients which have significant values, while others are rejected without any significant loss of information. The sparse signal is shown in Fig. 6 where only first 64 samples are displayed. These samples contain 99% energy of total samples. The remaining samples are not shown in the Fig. 6 due to their negligible energy content.

Regeneration of Original Sparse Signal At the receiver, the original sparse signal is recovered from its compressed form using sparse signal recovery algorithms, i.e. OMP, StOMP and gOMP. The performance of these algorithms is analysed at different levels of sparsity as shown in Fig. 7. As in OMP, the number of samples required to reconstruct the sparse signal is equal to sparsity measure K, whereas, in StOMP, the number of samples may fall below the sparsity value of the signal which degrades the performance very slightly. This happen due to the missed detections or false discoveries. False discoveries are also known as false alarms. The number of coordinates which are in the actual support of the signal but not appear in the selected coordinates is known as missed detections. However, false alarms are the number of incorrectly selected coordinates which are not in the support of signal but chosen in signal estimation. However, in gOMP, n samples are selected for each sparsity level, and after that, these samples are pruned to K for actual sparse signal estimation. Figure 7 shows that there is no noticeable difference in the signal recovery using all three algorithms. The performance of all algorithms improves as signal sparsity increases. The detailed analysis is presented in Tables 4,5 and 6 for OMP, StOMP and gOMP algorithms, respectively. The parameters considered for performance evaluation are RMSE, PSNR and execution time. There is no such a significant difference in the PSNR and RMSE values of recovered signal, but a large variation is observed in the number of iterations from OMP to StOMP and gOMP. In StOMP and gOMP, far less iterations are required than OMP which in turn affect their computational time. The execution time for each of the algorithm is tabulated in Table 7. In order to recover the sparse signal with the sparsity 8, the run time decreases from OMP to StOMP and gOMP and the difference in their execution time increases when sparsity increases to 128. In this way, StOMP and gOMP are more effective algorithms than OMP in terms of their accuracy and execution speed.



(a) (b) (c)

Fig. 7 Sparse signal recovery in WSN using $N = 1024$ original samples and signal reconstruction using $K = 8, \cdots$ 128 samples via **a** OMP, **b** StOMP and **c** gOMP, respectively

4 Conclusion

A temperature signal is measured using NI WSN thermocouple programmable input node at the same time interval. The signal is found highly temporally correlated; hence, it becomes enough sparse when represented into discrete cosine transform domain. The sparse signal is further compressed using deterministic binary block diagonal measurement matrix and transmitted to the receiver. At the receiver end, compressed signal is decoded into the original signal using greedy iterative algorithms. Here OMP, StOMP and gOMP algorithms are used for decoding purpose. A comparative analysis of all three algorithms is performed in terms of PSNR, RMSE and their computational time. The PSNR and RMSE values of the recovered signal using all three algorithms are found approximately equal. However, the large

Table 4 Performance parameters obtained in signal reconstruction via OMP

K	NS	CR	RMSE	PSNR [dB]
8	8	128	0.2604	43.7187
16	16	64	0.2392	44.4543
32	32	32	0.2017	45.9385
64	64	16	0.1688	47.4820
128	128	8	0.1441	48.8549

Table 5 Performance parameters obtained in signal reconstruction via StOMP

S	*K*	NS	CR	RMSE	PSNR [dB]
3	8	8	128	0.2670	43.4995
3	16	12	85	0.2490	44.1067
4	32	19	54	0.2445	44.2645
4	64	44	23	0.2067	45.7224
5	128	81	13	0.1623	47.8270

Table 6 Performance parameters obtained in signal reconstruction via gOMP

k	*K*	*n*	CR	RMSE	PSNR [dB]
4	8	8	128	0.2649	43.5702
5	16	16	64	0.2414	44.3760
5	32	32	32	0.2204	45.1670
7	64	64	16	0.1835	46.7595
7	128	128	8	0.1593	47.9883

Table 7 Simulation run time for OMP, StOMP and gOMP in milliseconds

K	8	16	32	64	128
OMP	0.8	2	5	20	200
StOMP	0.5	0.6	1	2	5
gOMP	1	3	6	10	12

difference in their execution time is observed. In order to recover the original signal using 8 samples, StOMP run 1.6 times faster than OMP, whereas gOMP run with approximately equal speed to that of OMP. In a typical case of 128 samples, the speed of recovery via StOMP and gOMP become 40 times and 17 times faster than OMP algorithm, respectively. These results establish the superior performance of StOMP and gOMP over OMP. Based on these results, this is recommended that StOMP and gOMP should be preferred for solving large-scale sparse problems like medical images such as magnetic resonance imaging, electrocardiogram signals and radar.

References

1. Needell, D.: Topics in compressed sensing. arXiv preprint arXiv:0905.4482 (2009)
2. Qaisar, S.; Bilal, R.M.; Iqbal, W.; Naureen, M.; Lee, S.: Compressive sensing: from theory to applications, a survey. J. Commun. Netw. **15**(5), 443–456 (2013)
3. Breen, P.: Algorithms for sparse approximation. *School of Mathematics, University of Edinburgh,4*. (2009). https://s3.amazonaws com/academia.edu.documents/42908668/file_5_52170211052012 .pdf?AWSAccessKeyId=AKIAIWOWYYGZ2Y53UL3A&Expir es=1504374516&Signature=pDMqbk%2BXS210FjqF3dGs7UcV HDQ%3D&response-content-disposition=inline%3B%20filename %3DAlgorithms_for_Sparse_Approximation.pdf
4. Donoho, D.L.: Compressed sensing. IEEE Trans. Inf. Theory **52**(4), 1289–1306 (2006)
5. Candès, E.J.; Wakin, M.B.: An introduction to compressive sampling. IEEE Signal Process. Mag. **25**(2), 21–30 (2008)
6. Patterson, S.; Eldar, Y. C.; Keidar, I.: Distributed sparse signal recovery for sensor networks. In: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 4494–4498 (2013).
7. Akyildiz, I.F.; Su, W.; Sankarasubramaniam, Y.; Cayirci, E.: Wireless sensor networks: a survey. Comput. Netw. **38**(4), 393–422 (2002)
8. Cheng, L.; Cao, J.; Chen, C.; Chen, H.; Ma, J.: Towards intelligent contention-based geographic forwarding in wireless sensor networks. IET Commun. **5**(11), 1711–1719 (2011)
9. Huang, P.; Chen, H.; Xing, G.; Tan, Y.: SGF: a state-free gradient-based forwarding protocol for wireless sensor networks. ACM Trans. Sens. Netw. (TOSN) **5**(2), 14 (2009)
10. Yue, Y.; Li, J.; Fan, H.; Qin, Q.: Optimization-based extreme learning machine for data fusion in mobile wireless sensor networks. Int. J. Innov. Comput. Inf. Control **12**(5), 1423–1438 (2016)
11. Li, H.; Zhao, Y.; Sun, Y.: Wireless sensor network based on high-dimensional quantum communication. Int. J. Innov. Comput Inf. Control **11**(6), 2119–2133 (2015)
12. Ravelomanantsoa, A.; Rabah, H.; Rouane, A.: Compressed sensing: A simple deterministic measurement matrix and a fast recovery algorithm. IEEE Trans. Instrum. Meas. **64**(11), 3405–3413 (2015)
13. Ravelomanantsoa, A.; Rabah, H.; Rouane, A.: Fast and efficient signals recovery for deterministic compressive sensing: Applications to biosignals. In: 2015 Conference on Design and Architectures for Signal and Image Processing (DASIP), pp. 1–6 (2015)
14. Baraniuk, R.G.: Compressive sensing. IEEE Signal Process. Mag. **24**(4), 118–121 (2007)
15. Davenport, M.A.; Duarte, M.F.; Eldar, Y.C.; Kutyniok, G.: Introduction to compressed sensing. In: Eldar, Y.C., Kutyniok, G. (eds.) Compressed Sensing: Theory and Applications, pp. 1–64. Cambridge University Press, Cambridge (2012)
16. Candes, E.J.; Romberg, J.K.; Tao, T.: Stable signal recovery from incomplete and inaccurate measurements. Commun. Pure Appl. Math. **59**(8), 1207–1223 (2006)
17. Tropp, J.A.: Greed is good: algorithmic results for sparse approximation. IEEE Trans. Inf. Theory **50**(10), 2231–2242 (2004)
18. Tropp, J.A.; Gilbert, A.C.: Signal recovery from random measurements via orthogonal matching pursuit. IEEE Trans. Inf. Theory **53**(11), 4655–4666 (2007)
19. Donoho, D.L.; Tsaig, Y.; Drori, I.; Starck, J.-L.: Sparse solution of underdetermined systems of linear equations by stagewise orthogonal matching pursuit. IEEE Trans. Inf. Theory **58**(2), 1094–1121 (2012)
20. Dai, W.; Milenkovic, O.: Subspace pursuit for compressive sensing signal reconstruction. IEEE Trans. Inf. Theory **55**(5), 2230–2249 (2009)

21. Needell, D.; Tropp, J.A.: CoSaMP: iterative signal recovery from incomplete and inaccurate samples. Appl. Comput. Harmonic Anal. **26**(3), 301–321 (2009)
22. Pilastri, A. L.; Tavares, J. M.: Reconstruction algorithms in compressive sensing: an overview. In: 11th Edition of the Doctoral Symposium in Informatics Engineering (DSIE‘ 16) (2016)
23. Wang, J.; Kwon, S.; Shim, B.: Generalized orthogonal matching pursuit. IEEE Trans. Signal Process. **60**(11), 6202–6216 (2012)
24. Liu, E.; Temlyakov, V.N.: The orthogonal super greedy algorithm and applications in compressed sensing. IEEE Trans. Inf. Theory **58**(4), 2040–2047 (2012)
25. Xu, Z.: The performance of orthogonal multi-matching pursuit under the restricted isometry property. J. Comput. Math. **33**(5), 495–516 (2015)
26. Sartipi, M.: Low-complexity distributed compression in wireless sensor networks. Data Compress. Conf. (DCC) **2012**, 227–236 (2012)
27. Quer, G.; Masiero, R.; Pillonetto, G.; Rossi, M.; Zorzi, M.: Sensing, compression, and recovery for WSNs: sparse signal modeling and monitoring framework. IEEE Trans. Wireless Commun. **11**(10), 3447–3461 (2012)
28. Razzaque, M.A.; Dobson, S.: Energy-efficient sensing in wireless sensor networks using compressed sensing. Sensors **14**(2), 2822–2859 (2014)
29. Do, T.T.; Tran, T.D.; Gan, L. : Fast compressive sampling with structurally random matrices. In: IEEE International Conference on Acoustics, Speech and Signal Processing 2008. ICASSP 2008, pp. 3369–3372 (2008)
30. Ravelomanantsoa, A.; Rabah, H.; Rouane, A.: Simple and efficient compressed sensing encoder for wireless body area network. IEEE Trans. Instrum. Meas. **63**(11), 2973–2982 (2014)
31. Mallat, S.G.; Zhang, Z.: Matching pursuits with time-frequency dictionaries. IEEE Trans. Signal Process. **41**(11), 3397–3415 (1993)
32. Shen, Y.; Li, B.; Pan, W.; Li, J.: Analysis of generalised orthogonal matching pursuit using restricted isometry constant. Electron. Lett. **50**(14), 1020–1022 (2014)
33. Li, B.; Shen, Y.; Wu, Z.; Li, J.: Sufficient conditions for generalized orthogonal matching pursuit in noisy case. Sig. Process. **108**, 111–123 (2015)